AMENDMENTS TO THE CLAIMS

The following listing of claims replaces all prior listings of claims:

Listing of Claims:

1. (Canceled)

2. (Canceled)

3. (Currently Amended) The method in accordance with claim 20 ~~2~~, wherein the message content comprises ~~includes~~ one or more data objects.

4. (Original) The method in accordance with claim 3, wherein unwrapping the message from the markup language file envelope includes deserializing the one or more data objects.

5. (Currently Amended) The method in accordance with claim 20 ~~1~~, wherein the message format is an Idoc message format.

6. (Currently Amended) The method in accordance with claim 20 ~~1~~, further comprising transmitting, at the application integration system, the converted message to the receiving application, when the sending and receiving applications have different message formats storing a copy of the message.

7. (Currently Amended) A computer-implemented method for transmitting a message from a sending application through an application integration system, the method comprising:

determining, at a routing module of an application integration system that is implemented on one or more processors, an identity of a receiving application of the message, the application integration system utilizing an application integration system file format;

determining, at a mapping module of the application integration system, a receiving application file format used by the receiving application;

sending the identity of the receiving application of the message and the file format used by the receiving application to the sending application in response to a polling received from the sending application at the application integration system;

~~when the file format used by the receiving application is identical to a file format used by the sending application,~~ wrapping the message in a markup language file envelope if the receiving application file format is identical to a sending application file format used by the sending application, the wrapping comprising adding a message header in the application integration system file format while leaving the message content unchanged and without mapping or converting the message content according to the application integration system file format; ~~and when the sending and receiving applications have different file formats~~

converting the ~~format of the~~ message to the application integration system file ~~a third~~ format ~~used by the application integration system~~ if the receiving application file format differs from the sending application file format; and

routing either the markup language file envelope with the message or the converted message through the ~~an~~ application integration system to the receiving application~~, the application integration system comprising the routing module to determine the receiving application and the mapping module to determine the file format of the receiving application~~.

8. (Original) The method in accordance with claim 7, wherein the markup language file envelope defines an XML envelope having as a payload one or more serialized data objects of the message.

9. (Currently Amended) The method in accordance with claim 7, wherein determining ~~a~~ the receiving application file format ~~used by the receiving application further includes~~ comprises the mapping module retrieving file format data from a directory.

10. (Currently Amended) The method in accordance with claim 7, wherein determining the identity of the ~~a~~ receiving application of the message includes retrieving receiving at the routing module application data from a directory based on the content of the message.

11. (Canceled)

12. (Canceled)

13. (Canceled)

14. (Canceled)

15. (Canceled)

16. (Canceled)

17. (Canceled)

18. (Canceled)

19. (New) The method in accordance with claim 8, further comprising unwrapping the message from the markup language file envelope by deserializing the one or more data objects.

20. (New) A method comprising:

polling, by a first adapter of a sending application, a routing module and a mapping module of an integration server, the sending application sending a message to a receiving application, the message comprising message content in an original message format, the integration server being implemented on one or more processors and utilizing one or more open standard file formats, the polling comprising a request for an identity of the receiving application from the routing module and for a receiving application file format used by the receiving application from the mapping module;

wrapping, at the first adapter, the message in a markup language file envelope that corresponds to one of the one or more open standard file formats of the integration server if the receiving application file format is identical to a sending application file format used by the sending application, the wrapping comprising adding a message header in the one of the one or more open standard file formats of the integration server while leaving the message content unchanged and without mapping or converting the message content to a file according to any of the one or more open standard file formats of the integration server;

sending the message in the markup language file envelope to a second adapter of the receiving application via one or more pipeline services on the integration server; the one or more pipeline services utilizing the message header for logical routing of the message;

unwrapping the message from the markup language file envelope at the second adapter; and

passing the message content to the receiving application substantially in the original message format.

21. (New) A system comprising:

an integration server implemented on one or more processors, the integration server comprising a routing module and a mapping module and utilizing one or more open standard file formats; and

a sending application comprising a first adapter, the first adapter polling the routing module and the mapping module to determine an identity of a receiving application for a message sent from the sending application and a receiving application file format used by the receiving application, the identity being determined from the routing module and the receiving application file format being determined from the mapping module, the message comprising message content in an original message format, the first adapter wrapping the message in a markup language file envelope that corresponds to one of the one or more open standard file formats of the integration server if the receiving application file format is identical to a sending application file format used by the sending application, the wrapping comprising adding a message header in the one of the one or more open standard file formats of the integration server while leaving the message content unchanged and without mapping or converting the message content according to any of the one or more open standard file formats of the integration server, the first adapter sending the message in the markup language file envelope to a second adapter of the receiving application via one or more pipeline services on the integration server; the one or more pipeline services utilizing the message header for logical routing of the message.

22. (New) The system in accordance with claim 21, wherein the sending application converts the message to one of the one or more open standard file formats of the integration server if the receiving application file format differs from the sending application file format.

23. (New) The method in accordance with claim 20, further comprising: capturing configuration-specific descriptions of a runtime system landscape to an integration directory, the runtime system landscape comprising all installed applications available for communication via the integration server according to a current business scenario, the capturing comprising reading, to the integration directory from an integration repository, configuration-specific message interface information for each installed application in the current business scenario as well as routing objects specified in the current business scenario, the integration repository comprising design-time descriptions of the interface information and routing objects for all available applications and business scenarios.

24. (New) The method in accordance with claim 23, further comprising: accessing the integration directory by the mapping module to determine mappings that define required transformations between message interfaces, message types, or data types in the current business scenario, the accessing of the integration repository occurring prior to the polling by the first adapter.

25. (New) The method in accordance with claim 23, further comprising: accessing the integration directory by the routing module to determine the routing objects that determine potential recieve4rs of messages to be distributed between applications according to the current business scenario, the accessing of the integration repository occurring prior to the polling by the first adapter.